Adatbázisrendszerek I. – 3. Gyakorlat

Filekezelés Java-ban -Eclipse

Készítsen egy mappát a **meghajtóra** a neve: **XY_neptunkod** – ebbe mentse el a feladatokat (XY – mindenkinek a monogramja).

A korábban elkészített mappába készítsen egy neptunkod_mai datum – ebbe helyezze el a mai feladatokat.

Az elkészült feladatokat csomagolja be és töltse fel a Classroom rendszerbe.

Készítsék el a következő feladatot Eclipse alkalmazás segítségével.

A feladat elkészítéséhez az **Elmélet: Java állományok kezelése**

A projekt neve: **vezetéknév**

A csomag neve: **package.meiit.vezeteknev**

1 feladat

Írjon programot, amely egész típusú adatokat beolvassa a szöveges **vezeteknev.txt** állományból, és kiszámítja az adatok összegét! Osztály neve**: OlvasXY**

Mentés: *neptunkod_3.1.java*

A futtatás eredménye:

```
Adatok száma = 2
0.adat = 10
1.adat = 20
Összeg: 30
```

2. feladat

Írjon egy programot, amely egész típusú adatokat ír a **vezeteknev.txt** állományba! osztály neve: **IrXY**

Mentés: *neptunkod_3.2.java*

Először meg kell adni a beírt adatok számát pl. 3 db egész számot szeretnék beírni, majd a következő sorba külön-külön a számokat. Ezeket a számok beírja a vezeteknev.txt nevű állományba egymás után.

A következő, hogy meg is jeleníti a konzolon.

A futás eredménye:

```
Adatok száma = 3
0.adat = 10
1.adat = 20
3.adat = 30

Adatok kiírása: vezetek.txt állományba
10
20
```

3. feladat

A szabvány billentyűzetről olvasson be sorokat, egészen a “end” szóig. A beolvasott sorokat írja ki egy szövegfile-ba. A szövegfile nevét a bevitel első sorában adja meg. Az így létrehozott, lezárt állományt utána nyissa meg és írja vissza a lementett szöveget nagybetűs formában.

Útmutató:

```
public void hf1 () {
   String sor;
   String[] szavak;
   int sorid = 0 ;
   try {
      BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
      BufferedWriter bw = null;

      while ( sorid >= 0) {
         sor =   br.readLine();
         if (sorid == 0) {
            bw = new BufferedWriter(new FileWriter(sor));
         } else {
            bw.write(sor);
            bw.newLine();
         }
         sorid = sorid + 1;
         szavak = sor.split(" ");
         for (String sz : szavak){
            System.out.println(sz+":");
            if (sz.compareTo("end") == 0 ) {
               br.close();
               sorid  = -1;
            }
         }
      }
      bw.close();
      System.out.println("Ok");
   } catch (Exception ee){
      ee.printStackTrace();
   }
}
```

Mentés: *neptunkod_3.3.java*

4. feladat

Az előbb létrehozott, lezárt állományt nyissa meg és írja vissza a lementett szöveget nagybetűs formában a képernyőre.

Útmutató

```
public void hf2 (String fnev) {
  String sor;
  String[] szavak;
  int sorid = 0 ;
  try {
     BufferedReader br = new BufferedReader(new FileReader(fnev));
     while ( (sor =   br.readLine()) != null) {
        System.out.println(sor.toUpperCase());
     }
     br.close();
     System.out.println("Ok");
  } catch (Exception ee){
     ee.printStackTrace();
  }
}
```

Mentés: *neptunkod_3.4.java*

5. feladat

Végezze el egy fájl tartalmának másolását egy másik fájlba. Másolás közben a számjegyeket cserélje le szöveges alakra, szóközökel határolva. A másoló függvény a fileneveket az argumentumában kapja meg.

Útmutató:

```
public void hf3 (String fnevbe, String fnevki) {
   String sor;
   String[] szavak;
   String[] k1 = { "1", "2", "3", "4", "5", "6", "7", "8", "9", "0" };
   String[] k2 = { " egy ", " kettő ", " három ", " négy ", " öt ", " hat ", " hét ",
                "nyolc ", "kilenc ", " nulla" };
   int sorid = 0 ;
   try {
      BufferedWriter bw = new BufferedWriter(new FileWriter(fnevki));
      BufferedReader br = new BufferedReader(new FileReader(fnevbe));

      while ( (sor =   br.readLine()) != null) {
         for (int i=0; i<10; i++){
            sor =  sor.replace(k1[i],k2[i]);
         }
         bw.write(sor);
         bw.newLine();
      }
      br.close();
      bw.close();


        System.out.println("Ok");
    } catch (Exception ee){
        ee.printStackTrace();
    }
 }
```

Mentés: *neptunkod_3.5.java*

6. feladat

Tároljon le auto (rendszam, tipus, ar) rekordokat egymás után egy bináris állományban, majd készítsen függvényt az i. rekord visszaolvasására.

Útmutató

```
public     class Auto implements Serializable {

    private static final long serialVersionUID = 1L;
    String rsz;
    String tipus;
    int ar;

    public Auto (String r, String t, int a){
       this.rsz = r;
       this.tipus = t;
       this.ar = a;
    }


}



public void hf4 () {
    String sor;
    Auto[] autoim = {new Auto("R11","Opel",333),new Auto("R12","Fiat",233),
                new Auto("R14","Skoda",364)};

    try {
       ObjectOutputStream kifile = new ObjectOutputStream(
          new FileOutputStream  ("Autok.dat")
       );
       for (Auto auto : autoim) {
          kifile.writeObject(auto);
       }
       kifile.close();

    } catch (Exception e) {


      e.printStackTrace();
      System.out.println ("File nyitasi hiba");
   }
    System.out.println ("OK");
}
```

Mentés: *neptunkod_3.6.java*

7. feladat

Készítsen programot, amely felőző autó nyilvántartóból kiírja a 300-nál drágább autók rendszámait.

Útmutató:

```
public void hf5 () {
   String sor;
   Auto ma;

   try {
      File fn = new File("Autok.dat");
      if (fn.exists()) {

         ObjectInputStream kifile = new ObjectInputStream(
            new FileInputStream  ("Autok.dat")
         );
         try {
            while (true) {
               ma = (Auto) kifile.readObject();
               if (ma.ar > 300) {
                  System.out.println("rendszam=" + ma.rsz);
               }
            }
         } catch (EOFException ee){
            ma = null;
         }
         kifile.close();
      }

   } catch (Exception e) {
      e.printStackTrace();
      System.out.println ("File nyitasi hiba");
   }
   System.out.println ("OK2");
}
```

Mentés: *neptunkod_3.7.java*

8. feladat

Készítsen programot, mely fel tud vinni személyeket (azonosító és név) bináris fájlba. Készítsen függvényt a) új rekordot létrehozatalára, b) létező rekord törlésére c) létező rekord módosítására.

Útmutató:

- használjon bináris file-t
- használjon saját osztályt szerkezetet
- törlésnél másolja át a maradó részt

Mentés: *neptunkod_3.8.java*

9. Feladat

Az autókat tároló adatfile-ban végezze el az alábbi lekérdezési műveleteket:

- Számítsa ki a fájlban eltárolt autók átlagárát.
- Kérdezze le az eltárolt piros autók darabszámát.
- Keresse meg a legdrágább autót a fájlban.

Útmutató:

- olvassa át az állományt rekordonként
- a szükséges adatokat emelje ki a rekordból
- végezze ez a szükséges számításokat memória változók segítségével.

Mentés: *neptunkod_3.9.java*